Licenciatura Engenharia Informática e Multimédia

Modelação e Simulação de Sistemas Naturais – MSSN

Relatório Projeto 1

Docente Paulo Vieira

Trabalho realizado por:

- Fábio Dias, nº 42921

- Jorge Silva, nº 44615

Introdução

Para este primeiro projeto, foi-nos pedido a realização do *Jogo Da Vida*, versão 23/3, com base nas classes *CellularAutomata* e *Cell*, desenvolvidas nos vídeos práticos.

Assim como o *Diffusion-Limited Aggregation*, simplificado como *DLA*, com base nas classes *Walker* e *DLA,* também desenvolvidas nos vídeos práticos.

Para cada um destes exercícios, existem as opções facultativas de diferentes implementações e variações destes exercícios.

1. Jogo Da Vida

Comecemos por explicar o que é o *Jogo da Vida*. O *Jogo da Vida* é um **autómato celular**, nomeadamente um **autómato celular binário (k = 2)** ou **Autómato Celular 2D**. Este jogo, desenvolvido pelo matemático britânico John Horton Conway, têm como objetivo reproduzir através de regras simples, as alterações e mudanças em grupos de seres vivos simulando comportamentos.

Para esta simulação poder funcionar é então necessário existir algumas regras, neste caso em concreto, para o *Jogo da vida* utilizamos um sistema de ***Vizinhança de Moore***.

Este sistema pode ser então configurado utilizando diferentes *Raios*, neste caso, utilizamos a *Vizinhanças de Moore* com raio 1 (**r = 1**), isto é, para cada célula será calculado uma regra que têm em conta os seus vizinhos.

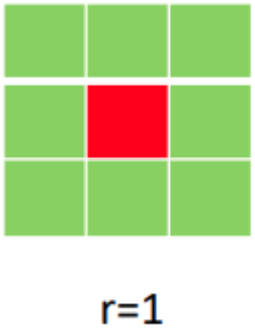

Figura 1 - Representação da Vizinhança de Moore

Após definirmos o raio de vizinhança, partimos então para as regras. Neste exercício é indicado que devemos utilizar a **regra 23/3**. Esta regra indica-nos então o seguinte:

a. Qualquer célula viva com menos de dois vizinhos, morre de solidão.
b. Qualquer célula viva com mais de três vizinhos vivos, morre de superpopulação.
c. Qualquer célula morta com exatamente três vizinhos vivos, renasce.
d. Qualquer célula com dois vizinhos vivos, permanece no mesmo estado.

Foi também implementado mais duas regras extras, a **regra 23/36** (conhecida como *HighLife)* e a **regra da Maioria**. A **regra 23/36** é muito idêntica a **regra 23/3** tendo como diferença que, para uma célula poder nascer precisa então só de ter 3 ou 6 vizinhos vivos.

A **regra da Maioria**, é uma regra que parte do princípio de que o próximo estado é definido caso a vizinhança esteja em maioria para um dos estados, isto é, vendo o próximo exemplo será mais fácil de perceber esta regra.

Figura 2 - Representação da Regra da Maioria.

Assumindo que na representação da imagem acima, a célula que está a ser calculada é a célula central. Podemos facilmente perceber que no universo dos vizinhos, o valor **0** é predominante, obrigando assim que na próxima iteração, essa célula passe do estado inicial 1 para o estado 0. Na mesma situação caso o valor predominante fosse 1, a célula a ser calculada iria então passar para o estado 1. No caso de existir a mesma quantidade de 0 (zeros) e 1 (uns) o valor irá manter-se.

Figura 3 - Representação da Regra da Maioria, com valores iguais.

Para a implementação deste exercício, usando por base o conhecimento adquirido pelas classes *CellularAutomata* e *Cell* desenvolvidas nas aulas práticas, criámos as nossas próprias classes.

GameOfLife.java - Esta classe, criada em conformidade com o que foi aprendido nas aulas práticas, é responsável pela inicialização da *board* que irá representar as células, a criação das células e a atribuição do tipo de regras implementadas para a simulação do Jogo da Vida.

GameCell.java – Esta classe representa uma célula que irá ser apresentada na *board*.

GameOfLifeDisplay.java – Esta classe é utilizada como controlador para representar as células no ecrã, assim como a *board*. Esta classe também possui um dos extras pedidos que é a introdução de um menu para podermos escolher o tipo de autómato que estamos a visualizar.

Temos então os seguintes resultados:

```
Bem vindos!
Comandos (teclas):
 - Para correr automáticamente: Barra de Espaços;
 - Para Iterar uma vez: Tecla 'B'
 - Para Reiniciar: Tecla 'R'
 - Para desenhar uma 'Shape': Tecla 'E'
 - Para mudar o tipo de Regra: Tecla 'T'
```

Figura 4 - Ao correr o Projeto irá ser apresentado este texto com indicações de teclas de atalho.

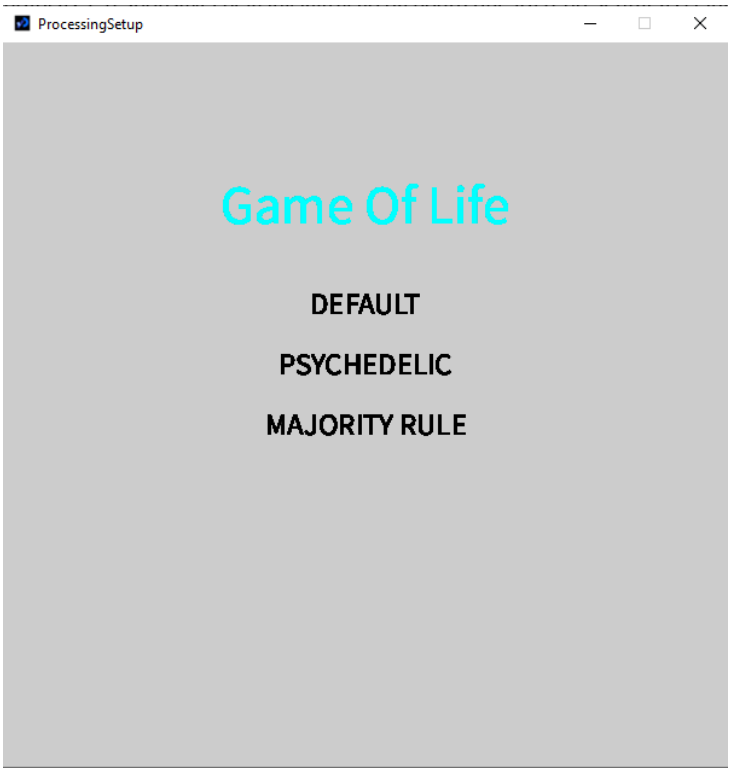

Figura 5 - Menu Inicial.

Neste menu podemos então escolher o modo *Default*, que irá apresentar a board com cores onde o utilizador pode desenhar um padrão ou, ao clicar na tecla 'E', irá ser apresentado um padrão automaticamente.

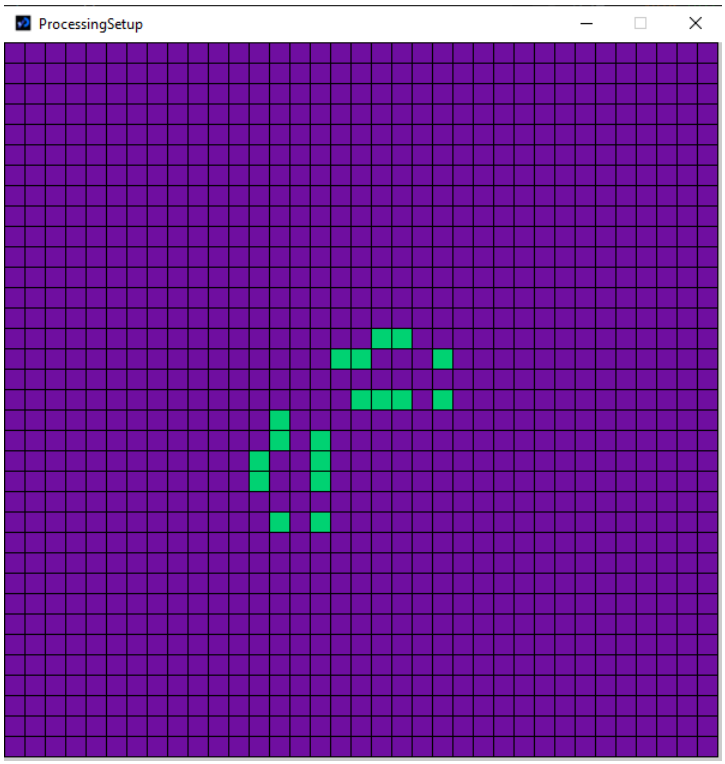

Figura 6 - Representação do Jogo da Vida (Default)

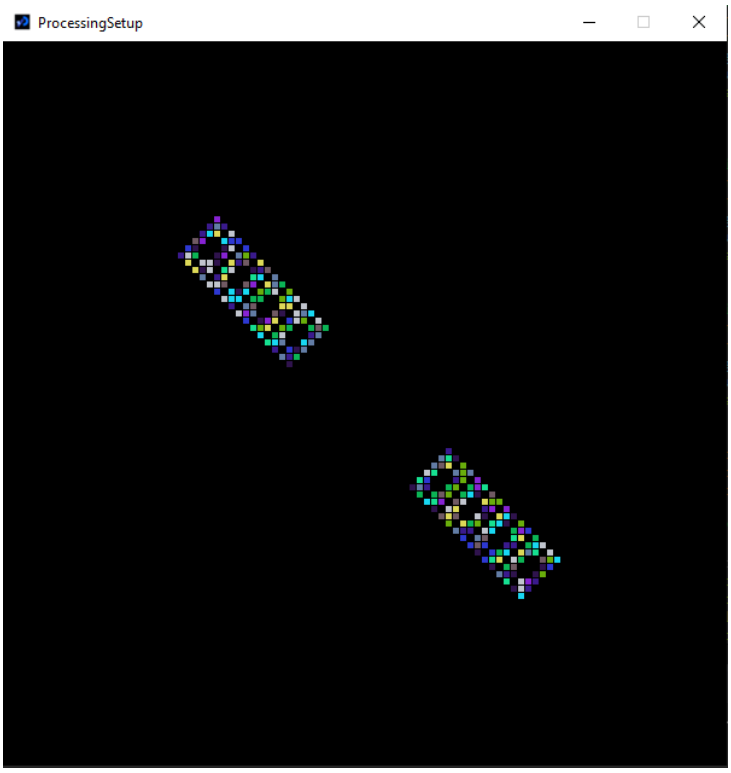

Figura 7 - Representação do Modo "Psychadelic" com a Regra 23/36 e um Padrão (Replicator).

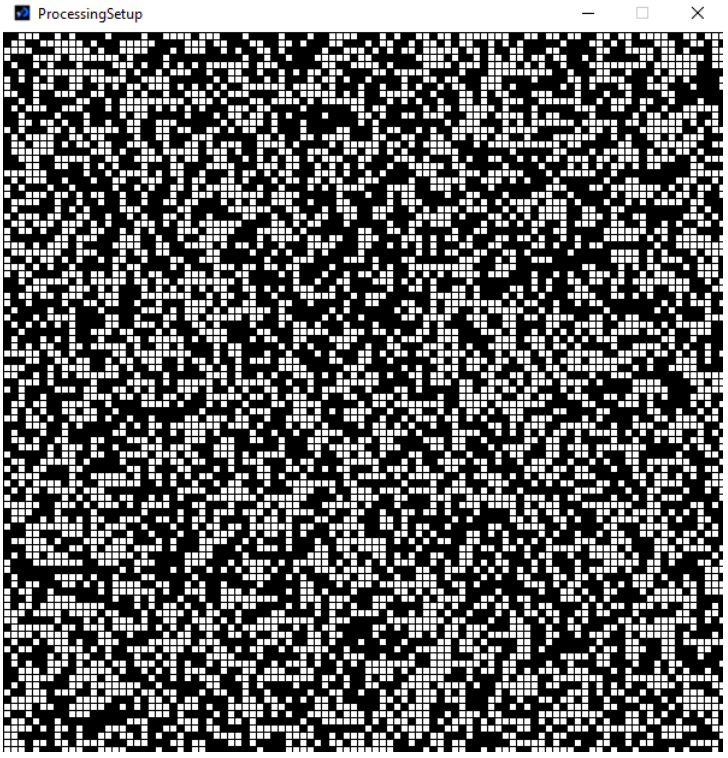

Figura 8 - Representação Inicial da Regra da Maioria.

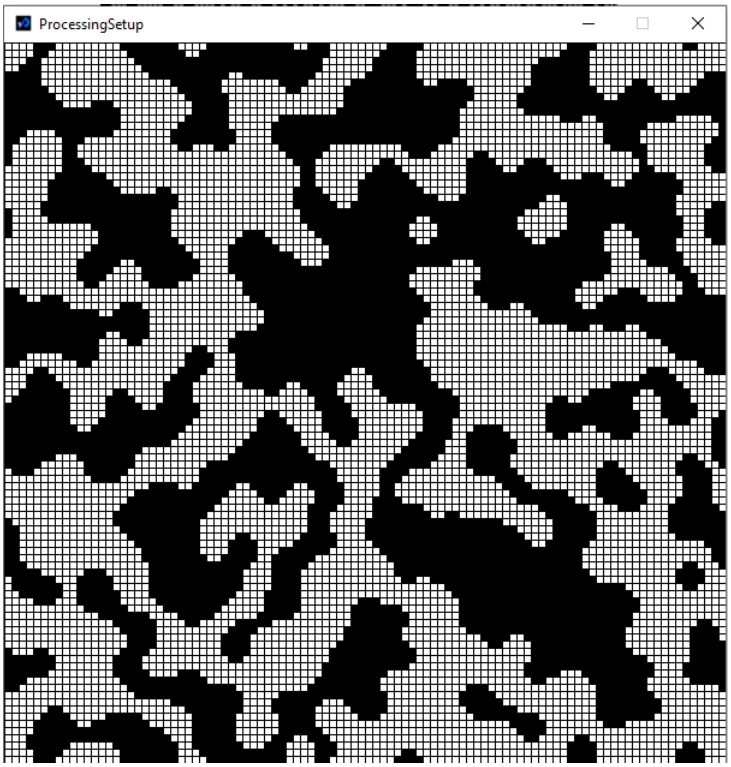

Figura 9 - Representação Final da Regra de Maioria.

2. DLA – Diffusion-Limited Aggregation

Neste exercício, tomámos por base as classes *DLA* e *Walker* implementadas nas vídeo-aulas práticas. Foram implementadas duas variações assim como o original. Para começar a simulação destes programas, basta carregar na barra de espaços, assim como para pausar a simulação.

Para o *DLA* original (Figura X), as únicas modificações foram o limite máximo de *Walkers* que seriam instanciados ao longo do programa, assim como o número máximo de *Walkers* ativos que não estivessem parados. Novos *Walkers* são instanciados quando, pelo menos, um previamente ativo, fica parado.

Deste forma atingimos uma melhor performance, dado que a simulação do movimento dos *Walkers* a ser calculado é muito menor do que se estivéssemos constantemente a instanciar novos *Walkers* até o número máximo definido.

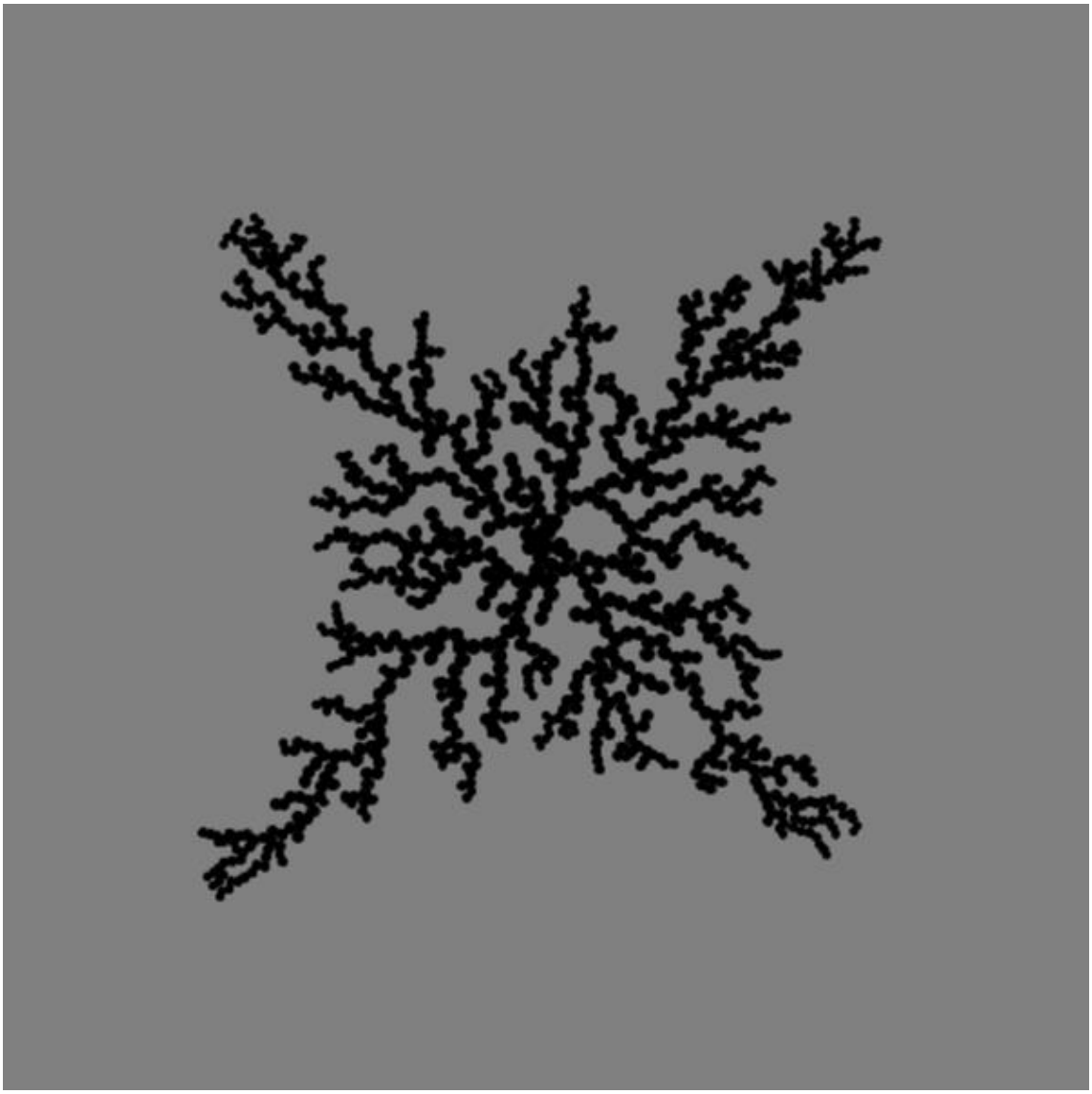

Figura 10 - DLA

Para a segunda variação, modificamos o ponto de criação dos *Walkers* para o topo da janela e, a direção para onde os encaminhamos, o fundo da janela. Modificámos as suas cores de forma a misturarem-se como a cor de fundo da janela, tornando-se assim "invisíveis".

Ao colidirem com esta linha criada no fundo da janela, param e ganham cor, dando a ideia de que esta estrutura semelhante a uma árvore, vai sendo criada lentamente não por *Walkers,* mas por mero crescimento da mesma.

O resultado final pode ser observado na Figura Y.

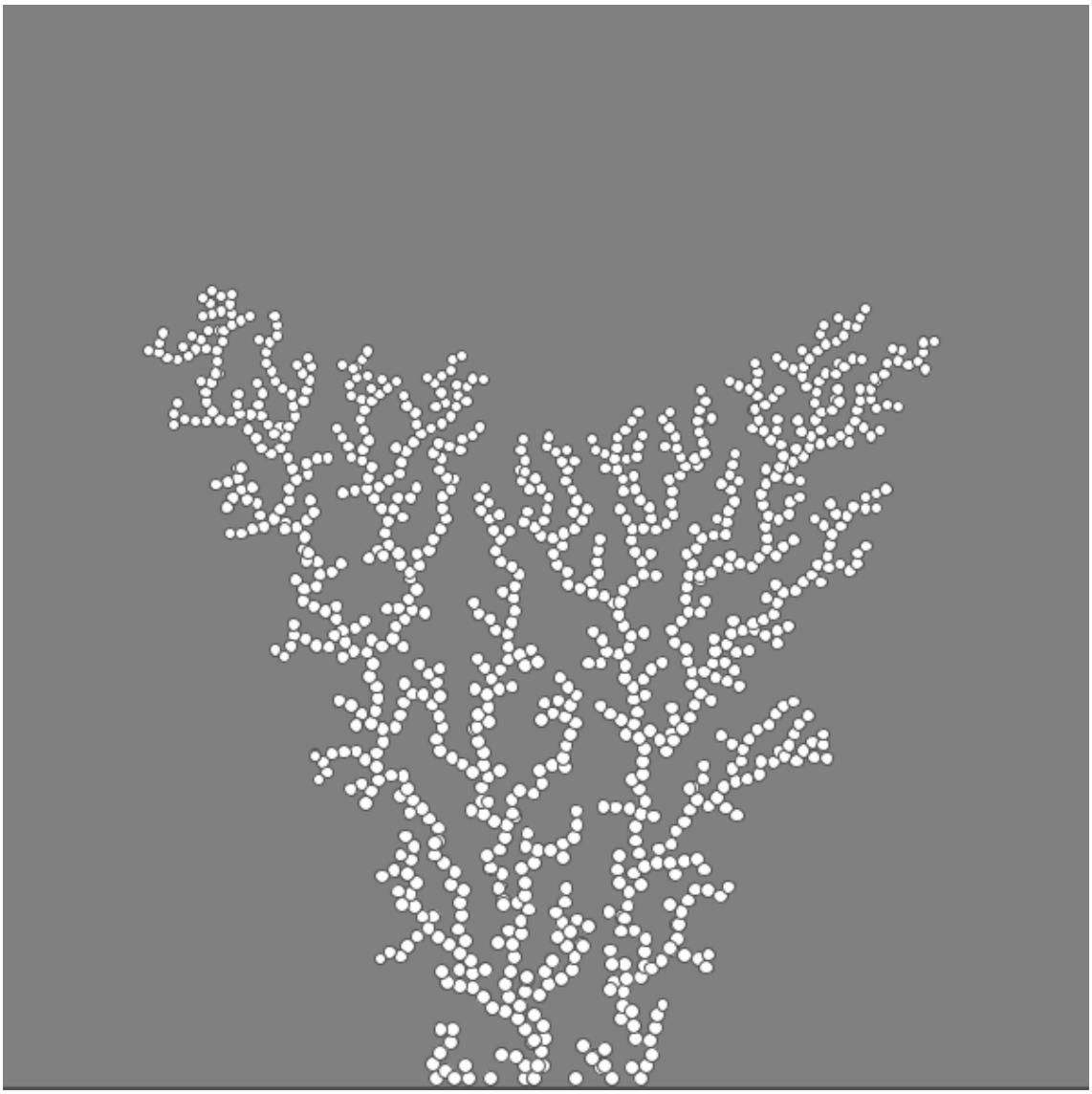

Figura 11 - DLAVariation1

Para a terceira e última variação, decidimos criar um círculo com um raio acentuado, mas com a mesma cor da janela de fundo, tornando-se assim impossível de distinguir do fundo. Os *Walkers* são instanciados no centro do ecrã, adicionalmente com um pequeno desvio, e são direccionados para as bordas do mesmo, onde existe o círculo mencionado previamente. Ao colidir com este, os *Walkers* param e passam a ter uma cor que vai sendo alterada conforme o número de *Walkers* parados, sendo assim um gradiente entre vermelho e um azul-claro, como demonstrado na Figura Z.

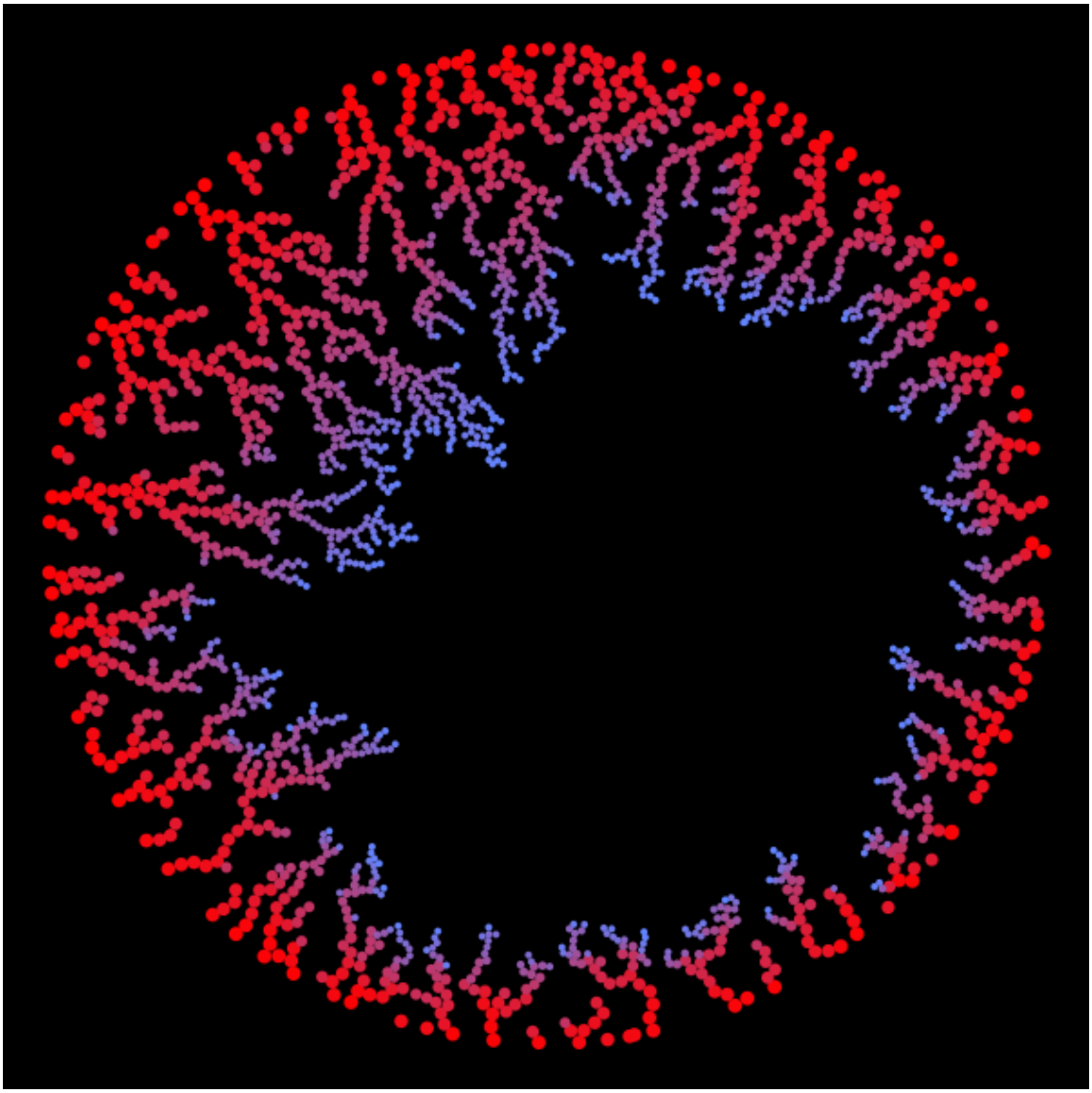

Figura 12 - DLAVariation2